What is Claimed:

1. A method of task management comprising the steps of:

a. receiving one or more tasks to be executed;

b. atomizing the one or more tasks into one or more atomic sub-tasks; and

c. designating access rights to one or more computing resources for each atomic sub-task of the one or more atomic sub-tasks.

2. The method according to claim 1, further comprising the step of:

d. scheduling the one or more atomic sub-tasks into a central task queue.

3. The method according to claim 2, wherein the step of scheduling the one or more atomic sub-tasks into a central task queue is done according to one or both of temporal and priority considerations.

4. The method according to claim 2, further comprising the step of:

e. obtaining from a first idle processor of a plurality of processors a first atomic sub-task from the central task queue, the first idle processor thereby inheriting the access rights to one or more computing resources of the first atomic sub-task in executing the first atomic sub-task.

5. The method according to claim 4, further comprising the step of:

f. obtaining from a further idle processor of the plurality of processors a further atomic sub-task from the central task queue, the further idle processor thereby inheriting the access rights to one or more computing resources of the further atomic sub-task in executing the further atomic sub-task.

6. The method according to claim 5, wherein steps e and f are repeated until there are no further idle processors or no further atomic sub-tasks in the central task queue.

7. The method according to claim 6, further comprising the step of combining one or more atomic results of execution of each atomic sub-task corresponding to a task into a result of the task.

8. A task-based library for processor management, comprising:

means for receiving tasks to be executed;

a task atomizer for atomizing the tasks into one or more atomic sub-tasks; and

an access rights generator for assigning protection attributes to the tasks designating access rights to one or more processing resources to be used in executing the tasks.

9. The task-based library of claim 8, further comprising a central task queue for storing the one or more atomic sub-tasks waiting to be executed.

10. The task-based library of claim 9, further comprising a task scheduler for arranging the one or more atomic sub-tasks in the central task queue.

11. The task-based library of claim 10, further comprising a combiner for combining execution results of the one or more atomic sub-tasks into an execution result of a task.

12. The task-based library of claim 11, further comprising one or more processors for executing the tasks.

13. The task-based library of claim 8, further comprising a combiner for combining execution results of the one or more atomic sub-tasks into an execution result of a task.

14. The task-based library of claim 13, further comprising one or more processors for executing the tasks.

15. The task-based library of claim 8, further comprising one or more processors for executing the tasks.

16. A method of task management comprising the steps of:

a. receiving one or more tasks to be executed;

b. atomizing the one or more tasks into one or more atomic sub-tasks;

c. designating access rights to one or more computing resources for each atomic sub-task of the one or more atomic sub-tasks;

d. scheduling the one or more atomic sub-tasks into a central task queue according to one or both of temporal and priority considerations;

e. obtaining via a first idle processor of a plurality of processors a first atomic sub-task from the central task queue, the first idle processor inheriting the access rights to one or more computing resources of the first atomic sub-task in executing the first atomic sub-task; and

f. obtaining via a further idle processor of the plurality of processors a further atomic sub-task from the central task queue, the further idle processor inheriting the access rights to one or more computing resources of the further atomic sub-task in executing the further atomic sub-task.

17. The method according to claim 16, further comprising the step of repeating steps e and f until there are no further idle processors or no further atomic sub-tasks in the central task queue.

18. The method according to claim 17, further comprising the step of combining one or more atomic results of execution of each atomic sub-task corresponding to a task into a result of the task.

19. The method according to claim 16, wherein the step d of scheduling the one or more atomic sub-tasks into a central task queue is done according to temporal considerations.

20. The method according to claim 16, wherein the step d of scheduling the one or more atomic sub-tasks into a central task queue is done according to priority considerations.